D 16

```
FFFFFFFFFFFFFFFF         111               111            AAAAAAAAA
FFFFFFFFFFFFFFFF         111               111            AAAAAAAAA
FFFFFFFFFFFFFFFF         111               111            AAAAAAAAA
FFF                   111111            111111         AAA          AAA
FFF                   111111            111111         AAA          AAA
FFF                   111111            111111         AAA          AAA
FFF                      111               111         AAA          AAA
FFF                      111               111         AAA          AAA
FFF                      111               111         AAA          AAA
FFFFFFFFFFFFF            111               111         AAA          AAA
FFFFFFFFFFFFF            111               111         AAA          AAA
FFFFFFFFFFFFF            111               111         AAA          AAA
FFF                      111               111         AAAAAAAAAAAAAAAA
FFF                      111               111         AAAAAAAAAAAAAAAA
FFF                      111               111         AAAAAAAAAAAAAAAA
FFF                      111               111         AAA          AAA
FFF                      111               111         AAA          AAA
FFF                      111               111         AAA          AAA
FFF                   111111111         111111111      AAA          AAA
FFF                   111111111         111111111      AAA          AAA
FFF                   111111111         111111111      AAA          AAA
```

C 4

```
**FILE**ID**ACPCNTRL
```

```
  AAAAAA      CCCCCCCC  PPPPPPPP      CCCCCCCC  NN      NN  TTTTTTTTTT  RRRRRRRR    LL
  AAAAAA      CCCCCCCC  PPPPPPPP      CCCCCCCC  NN      NN  TTTTTTTTTT  RRRRRRRR    LL
AA      AA  CC          PP      PP  CC          NN      NN      TT      RR      RR  LL
AA      AA  CC          PP      PP  CC          NN      NN      TT      RR      RR  LL
AA      AA  CC          PP      PP  CC          NNNN    NN      TT      RR      RR  LL
AA      AA  CC          PP      PP  CC          NNNN    NN      TT      RR      RR  LL
AA      AA  CC          PPPPPPPP    CC          NN  NN  NN      TT      RRRRRRRR    LL
AA      AA  CC          PPPPPPPP    CC          NN  NN  NN      TT      RRRRRRRR    LL
AAAAAAAAAA  CC          PP          CC          NN    NNNN      TT      RR  RR      LL
AAAAAAAAAA  CC          PP          CC          NN    NNNN      TT      RR  RR      LL
AA      AA  CC          PP          CC          NN      NN      TT      RR    RR    LL              ....
AA      AA  CC          PP          CC          NN      NN      TT      RR    RR    LL              ....
AA      AA    CCCCCCCC  PP            CCCCCCCC  NN      NN      TT      RR      RR  LLLLLLLLLL      ....
AA      AA    CCCCCCCC  PP            CCCCCCCC  NN      NN      TT      RR      RR  LLLLLLLLLL      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
ACP
V04-

; R
```

```
0 MODULE ACPCNTRL (
0                 LANGUAGE (BLISS32),
0                 IDENT = 'V04-000'
0                 ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  F11ACP Structure Level 1
1 !
1 ! ABSTRACT:
1 !
1 !       This module implements the ACP control I/O function.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  23-May-1979  17:07
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-004 STJ0310         Steven T. Jeffreys,     1-Jun-1982
1 !               Addd REMOUNT control function handler.  It's a NOP.
1 !
1 !       V03-003 LMP0026         L. Mark Pilant,         17-May-1982  14:15
1 !               Rearrange some code sequences to avoid the possibility of
1 !               taking a page fault at an elevated IPL.
1 !
```

```
;   58        0058  1  !        V03-002 ACG0285           Andrew C. Goldstein,     12-Apr-1982  17:26
;   59        0059  1  !                Fix cathedral window logic for empty headers
;   60        0060  1  !
;   61        0061  1  !        V03-001 LMP0018           L. Mark Pilant,          31-Mar-1982  12:00
;   62        0062  1  !                Modify to use a local of the window complete flag.  Also,
;   63        0063  1  !                fix som problems with remapping windows that don't map the
;   64        0064  1  !                beginning of the file.
;   65        0065  1  !
;   66        0066  1  !        V02-001 LMP0005           L. Mark Pilant,          29-Dec-1981  10:36
;   67        0067  1  !                Add routine to remap a file into multiple windows. This
;   68        0068  1  !                routine was taken, with minor modifications, from F11BACP.
;   69        0069  1  !
;   70        0070  1  !**
;   71        0071  1
;   72        0072  1
;   73        0073  1  LIBRARY 'SYS$LIBRARY:LIB.L32';
;   74        0074  1  REQUIRE 'SRC$:FCPDEF.B32';
;   75        0389  1
;   76        0390  1  !
;   77        0391  1  ! Range of control function codes recognized by this module.
;   78        0392  1  !
;   79        0393  1
;   80        0394  1  LITERAL
;   81        0395  1          MIN_CNTRLFUNC   = MINU (FIB$C_LOCK_VOL,
;   82        0396  1                                  FIB$C_UNLK_VOL,
;   83        0397  1                                  FIB$C_ENA_QUOTA,
;   84        0398  1                                  FIB$C_ADD_QUOTA,
;   85        0399  1                                  FIB$C_EXA_QUOTA,
;   86        0400  1                                  FIB$C_MOD_QUOTA,
;   87        0401  1                                  FIB$C_REM_QUOTA,
;   88        0402  1                                  FIB$C_DSA_QUOTA,
;   89        0403  1                                  FIB$C_REMAP
;   90        0404  1                                  ),
;   91        0405  1
;   92        0406  1          MAX_CNTRLFUNC   = MAXU (FIB$C_LOCK_VOL,
;   93        0407  1                                  FIB$C_UNLK_VOL,
;   94        0408  1                                  FIB$C_ENA_QUOTA,
;   95        0409  1                                  FIB$C_ADD_QUOTA,
;   96        0410  1                                  FIB$C_EXA_QUOTA,
;   97        0411  1                                  FIB$C_MOD_QUOTA,
;   98        0412  1                                  FIB$C_REM_QUOTA,
;   99        0413  1                                  FIB$C_DSA_QUOTA,
;  100        0414  1                                  FIB$C_REMAP
;  101        0415  1                                  );
;  102        0416  1
;  103        0417  1  FORWARD ROUTINE
;  104        0418  1          ACPCONTROL,                     ! ACPCONTROL function routine
;  105        0419  1          MARK_CATHEDRAL  : NOVALUE,      ! flag window as being cathedral
;  106        0420  1          ADD_@INDOW      : NOVALUE,      ! add a window to the queue
;  107        0421  1          REMOVE_WINDOW   : NOVALUE,      ! remove and deallocate a window segment
;  108        0422  1          LAST_SEGMENT    : NOVALUE;      ! set the window as the last segment
```

```
1 GLOBAL ROUTINE ACPCONTROL =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine implements the ACP control I/O function. It sets up
1 !       context and dispatches on the control function.
1 !
1 ! CALLING SEQUENCE:
1 !       ACPCONTROL ()
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       CLEANUP_FLAGS: cleanup action and status flags
1 !       IO_PACKET: address of I/O request packet
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       assorted status values
1 !
1 ! SIDE EFFECTS:
1 !       control function executed
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2       FIB             : REF BBLOCK,   ! address of user FIB
2       ABD             : REF BBLOCK,   ! address of buffer descriptor
2       STATUS;                         ! return status from called routine
2
2 EXTERNAL
2       CLEANUP_FLAGS   : BITVECTOR,    ! cleanup action and status flags
2       IO_PACKET       : REF BBLOCK;   ! address of caller's I/O packet
2
2 EXTERNAL ROUTINE
2       GET_FIB,                        ! get user FIB
2       REMAP_FILE      : NOVALUE;      ! remap the file into segmented windows
2
2
2 ! Set up control block pointers. If there is no complex buffer packet, then
2 ! this is an I/O kill call, which is a NOP.
2 !
2
2 IF NOT .IO_PACKET[IRP$V_COMPLX] THEN RETURN 1;
2
2 ABD = .BBLOCK [.IO_PACKET[IRP$L_SVAPTE], AIB$L_DESCRIPT];
2 FIB = GET_FIB (.ABD);
```

```
;  167          0480  2
;  168          0481  2 IF .BBLOCK [IO_PACKET[IRP$W_FUNC], IO$V_DMOUNT]
;  169          0482  2 THEN RETURN 1;                          ! DMOUNT is a NOP for ODS-1
;  170          0483  2
;  171          0484  2 IF .BBLOCK [IO_PACKET[IRP$W_FUNC], IO$V_REMOUNT]
;  172          0485  2 THEN RETURN 1;                          ! REMOUNT is a NOP for ODS-1
;  173          0486  2
;  174          0487  2 IF .FIB[FIB$W_CNTRLFUNC] EQL 0
;  175          0488  2 THEN RETURN 1;                          ! 0 is a NOP
;  176          0489  2
;  177          0490  2 ! Dispatch on the control function.
;  178          0491  2 !
;  179          0492  2
;  180          0493  2 IF .FIB[FIB$W_CNTRLFUNC] EQL FIB$C_REMAP THEN REMAP_FILE ();
;  181          0494  2
;  182          0495  2 RETURN 1;
;  183          0496  2
;  184          0497  1 END;                                    ! end of routine ACPCONTROL


                                                                        .TITLE  ACPCNTRL
                                                                        .IDENT  \V04-000\

                                                                        .EXTRN  CLEANUP_FLAGS, IO_PACKET
                                                                        .EXTRN  GET_FIB, REMAP_FILE

                                                                        .PSECT  $CODE$,NOWRT,2

                                               0000 00000               .ENTRY  ACPCONTROL, Save nothing                        ; 0423
                          50      0000G   CF   D0 00002                 MOVL    IO_PACKET, R0                                   ; 0476
          2A        2A    A0                03   E1 00007                 BBC     #3, 42(R0), 1$                                  ;
                          50         2C   B0   D0 0000C                 MOVL    @44(R0), ABD                                    ; 0478
                                           50   DD 00010                 PUSHL   ABD                                             ; 0479
                 0000G    CF                01   FB 00012                 CALLS   #1, GET_FIB                                     ;
                          51      0000G   CF   D0 00017                 MOVL    IO_PACKET, R1                                   ; 0481
          15        21    A1                02   E0 0001C                 BBS     #2, 33(R1), 1$                                  ;
          10        21    A1                03   E0 00021                 BBS     #3, 33(R1), 1$                                  ; 0484
                                     16     A0   B5 00026                 TSTW    22(FIB)                                         ; 0487
                                            0B   13 00029                 BEQL    1$                                              ;
                          10         16     A0   B1 0002B                 CMPW    22(FIB), #16                                    ; 0493
                                            05   12 0002F                 BNEQ    1$                                              ;
                 0000G    CF                00   FB 00031                 CALLS   #0, REMAP_FILE                                  ;
                          50                01   D0 00036 1$:             MOVL    #1, R0                                          ; 0495
                                                 04 00039                 RET                                                     ; 0497

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 0000
;  186       0498  1 GLOBAL ROUTINE REMAP_FILE : NOVALUE =
;  187       0499  1
;  188       0500  1 !++
;  189       0501  1 !
;  190       0502  1 ! FUNCTIONAL DESCRIPTION:
;  191       0503  1 !
;  192       0504  1 !       This routine is called when it becomes necessary to guarantee that
;  193       0505  1 !       the entire file is mapped.  This is done by creating, if necessary,
;  194       0506  1 !       multiple WCB's and linking them together.
;  195       0507  1 !
;  196       0508  1 ! CALLING SEQUENCE:
;  197       0509  1 !       REMAP_FILE ()
;  198       0510  1 !
;  199       0511  1 ! INPUT PARAMETERS:
;  200       0512  1 !       none
;  201       0513  1 !
;  202       0514  1 ! IMPLICIT INPUTS:
;  203       0515  1 !       PRIMARY_FCB: address of the current primary FCB
;  204       0516  1 !       CURRENT_WINDOW: address of the current primary window segment
;  205       0517  1 !
;  206       0518  1 ! OUTPUT PARAMETERS:
;  207       0519  1 !       none
;  208       0520  1 !
;  209       0521  1 ! IMPLICIT OUTPUTS:
;  210       0522  1 !       none
;  211       0523  1 !
;  212       0524  1 ! ROUTINE VALUE:
;  213       0525  1 !       none
;  214       0526  1 !
;  215       0527  1 ! SIDE EFFECTS:
;  216       0528  1 !       As many WCB's as are needed are allocated and linked to provide
;  217       0529  1 !       mapping for the extire file.  Any errors are noted for the user.
;  218       0530  1 !
;  219       0531  1 !--
;  220       0532  1
;  221       0533  2 BEGIN
;  222       0534  2
;  223       0535  2 LABEL
;  224       0536  2         HEADER_CHECK,                           ! loop to check window/header correspondence
;  225       0537  2         WINDOW_TRUNCATE;                        ! loop to match up last FCB with a window
;  226       0538  2
;  227       0539  2 LOCAL
;  228       0540  2         WINDOW_SEGMENT  : REF BBLOCK,   ! address of the next window segment
;  229       0541  2         OLD_WINDOW      : REF BBLOCK,   ! the original window
;  230       0542  2         NEW_WINDOW      : REF BBLOCK,   ! the new window list
;  231       0543  2         FCB             : REF BBLOCK,   ! address of the current FCB
;  232       0544  2         LAST_FCB        : REF BBLOCK,   ! address of the last FCB
;  233       0545  2         HEADER          : REF BBLOCK,   ! address of the header owned by an FCB
;  234       0546  2         MAP_AREA        : REF BBLOCK,   ! address of the map area in the header
;  235       0547  2         HEADER_VBN,                     ! current VBN in the header
;  236       0548  2         HEADER_COUNT,                   ! retrieval pointer count
;  237       0549  2         HEADER_LBN,                     ! retrieval pointer start LBN
;  238       0550  2         HEADER_POINTER  : REF BBLOCK,   ! pointer into map area
;  239       0551  2         NEXT_SEGMENT    : REF BBLOCK,   ! address of the segment after the next
;  240       0552  2         WINDOW_POINTER  : REF BBLOCK,   ! address of the window map area
;  241       0553  2         WINDOW_VBN,                     ! current VBN in the window
;  242       0554  2         WINDOW_ENDVBN;                  ! ending VBN of the window
```

```
2
2 EXTERNAL
2         CLEANUP_FLAGS   : BITVECTOR,    ! cleanup action and status flags
2         PRIMARY_FCB     : REF BBLOCK,   ! primary FCB for the file
2         CURRENT_WINDOW  : REF BBLOCK;   ! primary window for the file
2
2 EXTERNAL ROUTINE
2         DEALLOCATE,                     ! deallocate a block of memory
2         READ_HEADER,                    ! read a file header specified by FCB
2         TURN_WINDOW,                    ! map the file header specified
2         MARK_COMPLETE   : NOVALUE,      ! mark all windows as complete
2         MARK_INCOMPLETE;                ! mark all window segments as incomplete
2
2 ! Make sure that a file is there.
2 !
2
2 IF .CURRENT_WINDOW EQL 0 THEN ERR_EXIT (SS$_FILNOTACC);
2
2 ! Make sure it is actually necessary to do the remap operation.
2 !
2
2 IF .CURRENT_WINDOW[WCB$V_COMPLETE]
2 AND .CURRENT_WINDOW[WCB$V_CATHEDRAL]
2 THEN RETURN;
2
2 ! If there is a file accessed, try to build any necessary window segments.
2 ! There are three cases which can arise in trying to remap the entire file.
2 ! 1) The window completely maps the file but it was not required to; in this
2 ! case it is simply necessary to set WCB$V_CATHEDRAL. 2) The w'ndow was
2 ! previously complete, but is no longer due to an extension of the file; in
2 ! this case is is necessary to add the new window pointers to the last window
2 ! segment (which may be the primary window). 3) The file was never completely
2 ! mapped. In this case there are no special special conditions to consider.
2 ! All that is necessary is to traverse the linked FCB's to build the window
2 ! segments.
2 !
2
2 ! First case; WCB$V_COMPLETE is set. Simply set WCB$V_CATHEDRAL and return.
2 !
2
2 IF .CURRENT_WINDOW[WCB$V_COMPLETE] AND NOT .CURRENT_WINDOW[WCB$V_CATHEDRAL]
2 THEN
3     BEGIN
3     KERNEL_CALL (MARK_CATHEDRAL, .CURRENT_WINDOW);
3     RETURN;
2     END;
2
2 ! Second case; the file was previously mapped complete. Locate the FCB which
2 ! corresponds to the last window segment and start adding from there.
2 !
2
2 IF .CURRENT_WINDOW[WCB$V_CATHEDRAL]
2 THEN
3     BEGIN
3     WINDOW_SEGMENT = .CURRENT_WINDOW;
3     FCB = .PRIMARY_FCB;
3
```

```
3       UNTIL .WINDOW_SEGMENT[WCB$L_LINK] EQL 0
3       DO WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
3       NEW_WINDOW = .WINDOW_SEGMENT;                   ! remember current end point
3
3       WINDOW_ENDVBN = .WINDOW_SEGMENT[WCB$L_STVBN];
3       WINDOW_POINTER = .WINDOW_SEGMENT + WCB$C_MAP;
3       DECR J FROM .WINDOW_SEGMENT[WCB$W_NMAP] TO 1 DO
4           BEGIN
4           WINDOW_ENDVBN = .WINDOW_ENDVBN + .WINDOW_POINTER[WCB$W_COUNT];
4           WINDOW_POINTER = .WINDOW_POINTER + 6;
3           END;
3
4       HEADER_CHECK: BEGIN
4       LAST_FCB = .FCB;                                ! in case only 1 FCB
4       DO
5           BEGIN
5           IF .FCB[FCB$L_STVBN] GTR .WINDOW_ENDVBN THEN EXITLOOP 0;
5           LAST_FCB = .FCB;
5           FCB = .FCB[FCB$L_EXFCB];
5           END
4       UNTIL .FCB EQL 0;
4       FCB = .LAST_FCB;
4       HEADER = READ_HEADER (0, .FCB);
4       HEADER_VBN = .FCB[FCB$L_STVBN];
4       MAP_AREA = .HEADER + .HEADER[FH1$B_MPOFFSET]*2;
4       HEADER_POINTER = .MAP_AREA + FM1$C_POINTERS;
4       IF .WINDOW_ENDVBN EQL .HEADER_VBN THEN LEAVE HEADER_CHECK;
4
4       DECR J FROM .MAP_AREA[FM1$B_INUSE] / 2 TO 1
4       DO
5           BEGIN
5           HEADER_COUNT = .HEADER_POINTER[FM1$B_COUNT] + 1;        ! get count
5           HEADER_LBN = .HEADER_POINTER[FM1$W_LOWLBN];             ! low LBN
5           HEADER_LBN<16,8> = .HEADER_POINTER[FM1$B_HIGHLBN];      ! and high LBN
5           HEADER_POINTER = .HEADER_POINTER + 4;           ! update the map pointer
5           IF .WINDOW_ENDVBN GEQ .HEADER_VBN
5           AND .WINDOW_ENDVBN LSS .HEADER_VBN + .HEADER_COUNT
5           THEN LEAVE HEADER_CHECK;
5           HEADER_VBN = .HEADER_VBN + .HEADER_COUNT;
4           END;
4       FCB = .FCB[FCB$L_EXFCB];
4
4     ! The last VBN mapped does not have a corresponding FCB.  In this case it
4     ! is necessary to locate the window segment that corresponds to the last
4     ! FCB.
4     !
4
4       WINDOW_SEGMENT = .CURRENT_WINDOW;
5       WINDOW_TRUNCATE: BEGIN
5       DO
6           BEGIN
6           WINDOW_VBN = .WINDOW_SEGMENT[WCB$L_STVBN];
6           IF .WINDOW_VBN LEQ .HEADER_VBN THEN LEAVE WINDOW_TRUNCATE;
6           WINDOW_POINTER = .WINDOW_SEGMENT + WCB$C_MAP;
6           DECR J FROM .WINDOW_SEGMENT[WCB$W_NMAP] TO 1 DO
7               BEGIN
7               WINDOW_VBN = .WINDOW_VBN + .WINDOW_POINTER[WCB$W_COUNT];
```

```
7                WINDOW_POINTER = .WINDOW_POINTER + 6;
7                IF .WINDOW_VBN GEQ .HEADER_VBN THEN LEAVE WINDOW_TRUNCATE;
6                END;
6            WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
6            END
5        UNTIL .WINDOW_SEGMENT EQL 0;
5
5        BUG_CHECK (WCBFCBMNG, FATAL 'WCB/FCB correspondence broken');
5
4        END;                                    ! end of block WINDOW_TRUNCATE
4
4  ! The window which corresponds to the last FCB has been found.  Truncate the
4  ! current window and remove any succeeding window segments.
4  !
4
4        FCB = .LAST_FCB;
4        NEXT_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
4        KERNEL_CALL (LAST_SEGMENT, .WINDOW_SEGMENT);         ! current segment is now the end
4        UNTIL .NEXT_SEGMENT EQL 0
4        DO
5            BEGIN
5            LOCAL    JUNK_SEGMENT    : REF BBLOCK;   ! address of block to deallocate
5            JUNK_SEGMENT = .NEXT_SEGMENT;
5            NEXT_SEGMENT = .NEXT_SEGMENT[WCB$L_LINK];
5            KERNEL_CALL (REMOVE_WINDOW, .JUNK_SEGMENT);
4            END;
4
3        END;                                    ! end of block HEADER_CHECK
3
3  ! Map any additional file headers or rebuild the last window if cleaning up
3  ! from an extend operation.
3  !
3
3        WHILE 1 DO
4            BEGIN
4            KERNEL_CALL (TURN_WINDOW, .WINDOW_SEGMENT, .HEADER, 1, .FCB[FCB$L_STVBN]);
4            IF .CLEANUP_FLAGS[CLF_INCOMPLETE]
4            THEN
5                BEGIN
5                KERNEL_CALL (MARK_INCOMPLETE, .CURRENT_WINDOW);
5                ERR_EXIT (SS$_EXBYTLM);
4                END;
4            IF .FCB[FCB$L_EXFCB] EQL 0 THEN EXITLOOP 0;
4            UNTIL .WINDOW_SEGMENT[WCB$L_LINK] EQL 0
4            DO WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
4            FCB = .FCB[FCB$L_EXFCB];
4            HEADER = READ_HEADER (0, .FCB);
3            END;
3
3        WINDOW_SEGMENT = .NEW_WINDOW[WCB$L_LINK];
3        UNTIL .WINDOW_SEGMENT EQL 0
3        DO
4            BEGIN
4            KERNEL_CALL (ADD_WINDOW, .WINDOW_SEGMENT, .PRIMARY_FCB[FCB$L_WLBL]);
4            WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
3            END;
3
```

```
3        KERNEL_CALL (MARK_COMPLETE, .CURRENT_WINDOW);
3        RETURN;
2        END;
2
2    ! Third case; the file was never completely mapped. For this case no special
2    ! precautions need to be taken. Simply loop through all the FCB's associated
2    ! with the file, and create as many window segments as necessary.
2    !
2
2    FCB = .PRIMARY_FCB;
2    WINDOW_SEGMENT = .CURRENT_WINDOW;
2    KERNEL_CALL (MARK_CATHEDRAL, .WINDOW_SEGMENT);  !build cathedral windows
2
2    ! Now build the new windows using the original primary window as the base
2    ! for the new window segments.  This is necessary to aviod having to mung
2    ! the primary window address which may reside in several places.  It also
2    ! means that if an error occurs, the new window created will be valid, but
2    ! it will not be the same as it started out.
2    !
2
2    UNTIL .FCB EQL 0
2    DO
3        BEGIN
3        HEADER = READ_HEADER (0, .FCB);
3        UNTIL .WINDOW_SEGMENT[WCB$L_LINK] EQL 0
3        DO WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
3        KERNEL_CALL (TURN_WINDOW, .WINDOW_SEGMENT, .HEADER, 1, .FCB[FCB$L_STVBN]);
3        IF .CLEANUP_FLAGS[CLF_INCOMPLETE]
3        THEN
4            BEGIN
4            KERNEL_CALL (MARK_INCOMPLETE, .CURRENT_WINDOW);
4            ERR_EXIT (SS$_EXBYTLM);
3            END;
3        FCB = .FCB[FCB$L_EXFCB];
2        END;
2
2    WINDOW_SEGMENT = .CURRENT_WINDOW[WCB$L_LINK];
2    UNTIL .WINDOW_SEGMENT EQL 0
2    DO
3        BEGIN
3        KERNEL_CALL (ADD_WINDOW, .WINDOW_SEGMENT, .PRIMARY_FCB[FCB$L_WLBL]);
3        WINDOW_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
2        END;
2
2    KERNEL_CALL (MARK_COMPLETE, .CURRENT_WINDOW);
2    RETURN;
2
1    END;                                        ! end of routine REMAP_FILE


                                                             .EXTRN  PRIMARY_FCB, CURRENT_WINDOW
                                                             .EXTRN  DEALLOCATE, READ_HEADER
                                                             .EXTRN  TURN_WINDOW, MARK_COMPLETE
                                                             .EXTRN  MARK_INCOMPLETE
                                                             .EXTRN  SYS$CMKRNL, BUG$_WCBFCBMNG
```

```
                                  0FFC    00000         .ENTRY  REMAP_FILE, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 0498
                                                                R10,RT1                                     ;
                    5E              04 C2 00002         SUBL2   #4, SP                                      ;
                             0000G  CF D5 00005         TSTL    CURRENT_WINDOW                              ; 0571
                                    05 12 00009         BNEQ    1$                                          ;
                              00AC  8F BF 0000B         CHMU    #172                                        ;
                                       04 0000F         RET                                                 ;
                    50       0000G  CF D0 00010 1$:     MOVL    CURRENT_WINDOW, R0                          ; 0576
1D          0B      A0              05 E1 00015         BBC     #5, 11(R0), 3$                              ;
01          0B      A0              06 E1 0001A         BBC     #6, 11(R0), 2$                              ; 0577
                                       04 0001F         RET                                                 ;
12          0B      A0              05 E1 00020 2$:     BBC     #5, 11(R0), 3$                              ; 0595
0D          0B      A0              06 E0 00025         BBS     #6, 11(R0), 3$                              ;
                                    50 DD 0002A         PUSHL   R0                                          ; 0598
                                    01 DD 0002C         PUSHL   #1                                          ;
                                    5E DD 0002E         PUSHL   SP                                          ;
                             0000V  CF 9F 00030         PUSHAB  MARK_CATHEDRAL                              ;
                                  0224 31 00034         BRW     36$                                         ;
                    50       0000G  CF D0 00037 3$:     MOVL    CURRENT_WINDOW, R0                          ; 0606
03          0B      A0              06 E0 0003C         BBS     #6, 11(R0), 4$                              ;
                                  016F 31 00041         BRW     27$                                         ;
                    52              50 D0 00044 4$:     MOVL    R0, WINDOW_SEGMENT                          ; 0609
                    55       0000G  CF D0 00047         MOVL    PRIMARY_FCB, FCB                            ; 0610
                                20  A2 D5 0004C 5$:     TSTL    32(WINDOW_SEGMENT)                          ; 0612
                                    06 13 0004F         BEQL    6$                                          ;
                    52          20  A2 D0 00051         MOVL    32(WINDOW_SEGMENT), WINDOW_SEGMENT          ; 0613
                                    F5 11 00055         BRB     5$                                          ;
                    6E              52 D0 00057 6$:     MOVL    WINDOW_SEGMENT, NEW_WINDOW                  ; 0614
                    56          2C  A2 D0 0005A         MOVL    44(WINDOW_SEGMENT), WINDOW_ENDVBN           ; 0616
                    54          30  A2 9E 0005E         MOVAB   48(R2), WINDOW_POINTER                      ; 0617
                    50          16  A2 3C 00062         MOVZWL  22(WINDOW_SEGMENT), J                       ; 0618
                                    50 D6 00066         INCL    J                                           ;
                                    09 11 00068         BRB     8$                                          ;
                    51              84 3C 0006A 7$:     MOVZWL  (WINDOW_POINTER)+, R1                       ; 0620
                    56              51 C0 0006D         ADDL2   R1, WINDOW_ENDVBN                           ;
                    54              04 C0 00070         ADDL2   #4, WINDOW_POINTER                          ; 0621
                    F4              50 F5 00073 8$:     SOBGTR  J, 7$                                       ; 0618
                    5A              55 D0 00076         MOVL    FCB, LAST_FCB                               ; 0625
                    56          2C  A5 D1 00079 9$:     CMPL    44(FCB), WINDOW_ENDVBN                      ; 0628
                                    09 14 0007D         BGTR    10$                                         ;
                    5A              55 D0 0007F         MOVL    FCB, LAST_FCB                               ; 0629
                    55          0C  A5 D0 00082         MOVL    12(FCB), FCB                                ; 0630
                                    F1 12 00086         BNEQ    9$                                          ; 0632
                    55              5A D0 00088 10$:    MOVL    LAST_FCB, FCB                               ; 0633
                                    55 DD 0008B         PUSHL   FCB                                         ; 0634
                                    7E D4 0008D         CLRL    -(SP)                                       ;
             0000G  CF              02 FB 0008F         CALLS   #2, READ_HEADER                             ;
                    58              50 D0 00094         MOVL    R0, HEADER                                  ;
                    53          2C  A5 D0 00097         MOVL    44(FCB), HEADER_VBN                         ; 0635
                    50          01  A8 9A 0009B         MOVZBL  1(HEADER), R0                               ; 0636
                    51            6840 3E 0009F         MOVAW   (HEADER)[R0], MAP_AREA                      ;
                    50          0A  A1 9E 000A3         MOVAB   10(R1), HEADER_POINTER                      ; 0637
                    53              56 D1 000A7         CMPL    WINDOW_ENDVBN, HEADER_VBN                   ; 0638
                                    03 12 000AA         BNEQ    11$                                         ;
                                  0096 31 000AC         BRW     20$                                         ;
                    59          08  A1 9A 000AF 11$:    MOVZBL  8(MAP_AREA), R9                             ; 0640
                    59              02 C6 000B3         DIVL2   #2, R9                                      ;
```

```
                                             59 D6 000B6        INCL     J                                        ;
                                             20 11 000B8        BRB      14$                                      ;
                               57      01    A0 9A 000BA 12$:   MOVZBL   1(HEADER_POINTER), HEADER_COUNT          ; 0643
                                             57 D6 000BE        INCL     HEADER_COUNT                             ;
                               5B      02    A0 3C 000C0        MOVZWL   2(HEADER_POINTER), HEADER_LBN            ; 0644
      5B       08              10            80 F0 000C4        INSV     (HEADER_POINTER)+, #16, #8, HEADER_LBN   ; 0645
                               53            56 D1 000C9        CMPL     WINDOW_ENDVBN, HEADER_VBN                ; 0647
                                             09 19 000CC        BLSS     13$                                      ;
               51              53            57 C1 000CE        ADDL3    HEADER_COUNT, HEADER_VBN, R1             ; 0648
                               51            56 D1 000D2        CMPL     WINDOW_ENDVBN, R1                        ;
                                             6E 19 000D5        BLSS     20$                                      ;
                               53            57 C0 000D7 13$:   ADDL2    HEADER_COUNT, HEADER_VBN                 ; 0650
                               DD            59 F5 000DA 14$:   SOBGTR   J, 12$                                   ; 0640
                               55      0C    A5 D0 000DD        MOVL     12(FCB), FCB                             ; 0652
                               52   0000G    CF D0 000E1        MOVL     CURRENT_WINDOW, WINDOW_SEGMENT           ; 0659
                               51      2C    A2 D0 000E6 15$:   MOVL     44(WINDOW_SEGMENT), WINDOW_VBN           ; 0663
                               53            51 D1 000EA        CMPL     WINDOW_VBN, HEADER_VBN                   ; 0664
                                             27 15 000ED        BLEQ     18$                                      ;
                               54      30    A2 9E 000EF        MOVAB    48(R2), WINDOW_POINTER                   ; 0665
                               50      16    A2 3C 000F3        MOVZWL   22(WINDOW_SEGMENT), J                    ; 0666
                                             50 D6 000F7        INCL     J                                        ;
                                             0E 11 000F9        BRB      17$                                      ;
                               56            84 3C 000FB 16$:   MOVZWL   (WINDOW_POINTER)+, R6                    ; 0668
                               51            56 C0 000FE        ADDL2    R6, WINDOW_VBN                           ;
                               54            04 C0 00101        ADDL2    #4, WINDOW_POINTER                       ; 0669
                               53            51 D1 00104        CMPL     WINDOW_VBN, HEADER_VBN                   ; 0670
                                             0D 18 00107        BGEQ     18$                                      ;
                               EF            50 F5 00109 17$:   SOBGTR   J, 16$                                   ; 0666
                               52      20    A2 D0 0010C        MOVL     32(WINDOW_SEGMENT), WINDOW_SEGMENT       ; 0672
                                             D4 12 00110        BNEQ     15$                                      ; 0674
                                           FEFF    00112        BUGW                                              ; 0676
                                          0000*    00114        .WORD    <BUG$_WCBFCBMNG!4>                       ;
                               55            5A D0 00116 18$:   MOVL     LAST_FCB, FCB                            ; 0684
                               54      20    A2 D0 00119        MOVL     32(WINDOW_SEGMENT), NEXT_SEGMENT         ; 0685
                                             52 DD 0011D        PUSHL    WINDOW_SEGMENT                           ; 0686
                                             01 DD 0011F        PUSHL    #1                                       ;
                                             5E DD 00121        PUSHL    SP                                       ;
                                    0000V    CF 9F 00123        PUSHAB   LAST_SEGMENT                             ;
                   00000000G   9F            04 FB 00127 19$:   CALLS    #4, @#SYS$CMKRNL                         ;
                                             54 D5 0012E        TSTL     NEXT_SEGMENT                             ; 0687
                                             13 13 00130        BEQL     20$                                      ;
                               50            54 D0 00132        MOVL     NEXT_SEGMENT, JUNK_SEGMENT               ; 0691
                               54      20    A4 D0 00135        MOVL     32(NEXT_SEGMENT), NEXT_SEGMENT           ; 0692
                                             50 DD 00139        PUSHL    JUNK_SEGMENT                             ; 0693
                                             01 DD 0013B        PUSHL    #1                                       ;
                                             5E DD 0013D        PUSHL    SP                                       ;
                                    0000V    CF 9F 0013F        PUSHAB   REMOVE_WINDOW                            ;
                                             E2 11 00143        BRB      19$                                      ;
                                       2C    A5 DD 00145 20$:   PUSHL    44(FCB)                                  ; 0704
                                             01 DD 00148        PUSHL    #1                                       ;
                                     0104    8F BB 0014A        PUSHR    #^M<R2,R8>                               ;
                                             04 DD 0014E        PUSHL    #4                                       ;
                                             5E DD 00150        PUSHL    SP                                       ;
                                    0000G    CF 9F 00152        PUSHAB   TURN_WINDOW                              ;
                   00000000G   9F            07 FB 00156        CALLS    #7, @#SYS$CMKRNL                         ;
               03      0000G   CF            02 E1 0015D        BBC      #2, CLEANUP_FLAGS+1, 21$                 ; 0705
                                           00A1 31 00163        BRW      31$                                      ;
```

```
                       0C    A5  D5 00166 21$:    TSTL    12(FCB)                                 ; 0711
                             1D  13 00169         BEQL    24$                                     ;
                       20    A2  D5 0016B 22$:    TSTL    32(WINDOW_SEGMENT)                      ; 0712
                             06  13 0016E         BEQL    23$                                     ;
               52      20    A2  D0 00170         MOVL    32(WINDOW_SEGMENT), WINDOW_SEGMENT      ; 0713
                             F5  11 00174         BRB     22$                                     ;
               55      0C    A5  D0 00176 23$:    MOVL    12(FCB), FCB                            ; 0714
                             55  DD 0017A         PUSHL   FCB                                     ; 0715
                             7E  D4 0017C         CLRL    -(SP)                                   ;
        0000G  CF                02  FB 0017E         CALLS   #2, READ_HEADER                         ;
               58            50  D0 00183         MOVL    R0, HEADER                              ;
                             BD  11 00186         BRB     20$                                     ; 0702
50             6E            20  C1 00188 24$:    ADDL3   #32, NEW_WINDOW, R0                     ; 0718
               52            60  D0 0018C         MOVL    (R0), WINDOW_SEGMENT                    ;
                             03  12 0018F 25$:    BNEQ    26$                                     ; 0719
                           00BB  31 00191         BRW     35$                                     ;
               50   0000G    CF  D0 00194 26$:    MOVL    PRIMARY_FCB, R0                         ; 0722
                       14    A0  DD 00199         PUSHL   20(R0)                                  ;
                             52  DD 0019C         PUSHL   WINDOW_SEGMENT                          ;
                             02  DD 0019E         PUSHL   #2                                      ;
                             5E  DD 001A0         PUSHL   SP                                      ;
                    0000V    CF  9F 001A2         PUSHAB  ADD_WINDOW                              ;
   000000000G  9F            05  FB 001A6         CALLS   #5, @#SYS$CMKRNL                        ;
               52      20    A2  D0 001AD         MOVL    32(WINDOW_SEGMENT), WINDOW_SEGMENT      ; 0723
                             DC  11 001B1         BRB     25$                                     ; 0719
               55   0000G    CF  D0 001B3 27$:    MOVL    PRIMARY_FCB, FCB                        ; 0735
               52   0000G    CF  D0 001B8         MOVL    CURRENT_WINDOW, WINDOW_SEGMENT          ; 0736
                             52  DD 001BD         PUSHL   WINDOW_SEGMENT                          ; 0737
                             01  DD 001BF         PUSHL   #1                                      ;
                             5E  DD 001C1         PUSHL   SP                                      ;
                    0000V    CF  9F 001C3         PUSHAB  MARK_CATHEDRAL                          ;
   000000000G  9F            04  FB 001C7         CALLS   #4, @#SYS$CMKRNL                        ;
                             55  D5 001CE 28$:    TSTL    FCB                                     ; 0746
                             53  13 001D0         BEQL    33$                                     ;
                             55  DD 001D2         PUSHL   FCB                                     ; 0749
                             7E  D4 001D4         CLRL    -(SP)                                   ;
        0000G  CF            02  FB 001D6         CALLS   #2, READ_HEADER                         ;
               58            50  D0 001DB         MOVL    R0, HEADER                              ;
                       20    A2  D5 001DE 29$:    TSTL    32(WINDOW_SEGMENT)                      ; 0750
                             06  13 001E1         BEQL    30$                                     ;
               52      20    A2  D0 001E3         MOVL    32(WINDOW_SEGMENT), WINDOW_SEGMENT      ; 0751
                             F5  11 001E7         BRB     29$                                     ;
                       2C    A5  DD 001E9 30$:    PUSHL   44(FCB)                                 ; 0752
                             01  DD 001EC         PUSHL   #1                                      ;
                     0104    8F  BB 001EE         PUSHR   #^M<R2,R8>                              ;
                             04  DD 001F2         PUSHL   #4                                      ;
                             5E  DD 001F4         PUSHL   SP                                      ;
                    0000G    CF  9F 001F6         PUSHAB  TURN_WINDOW                             ;
   000000000G  9F            07  FB 001FA         CALLS   #7, @#SYS$CMKRNL                        ;
18      0000G  CF            02  E1 00201         BBC     #2, CLEANUP_FLAGS+1, 32$                ; 0753
                    0000G    CF  DD 00207 31$:    PUSHL   CURRENT_WINDOW                          ; 0756
                             01  DD 0020B         PUSHL   #1                                      ;
                             5E  DD 0020D         PUSHL   SP                                      ;
                    0000G    CF  9F 0020F         PUSHAB  MARK_INCOMPLETE                         ;
   000000000G  9F            04  FB 00213         CALLS   #4, @#SYS$CMKRNL                        ;
                     2A14    8F  BF 0021A         CHMU    #10772                                  ; 0757
                                 04 0021E         RET                                             ;
```

```
                        55        0C  A5  D0 0021F 32$:     MOVL    12(FCB), FCB                               ; 0759
                                      A9  11 00223          BRB     28$                                        ; 0746
                        50     0000G  CF  D0 00225 33$:     MOVL    CURRENT_WINDOW, R0                         ; 0762
                        52        20  A0  D0 0022A          MOVL    32(R0), WINDOW_SEGMENT                     ;
                                      1F  13 0022E 34$:     BEQL    35$                                        ; 0763
                        50     0000G  CF  D0 00230          MOVL    PRIMARY_FCB, R0                            ; 0766
                                  14  A0  DD 00235          PUSHL   20(R0)                                     ;
                                      52  DD 00238          PUSHL   WINDOW_SEGMENT                             ;
                                      02  DD 0023A          PUSHL   #2                                         ;
                                      5E  DD 0023C          PUSHL   SP                                         ;
                               0000V  CF  9F 0023E          PUSHAB  ADD_WINDOW                                 ;
              00000000G  9F           05  FB 00242          CALLS   #5, @#SYS$CMKRNL                           ;
                        52        20  A2  D0 00249          MOVL    32(WINDOW_SEGMENT), WINDOW_SEGMENT         ; 0767
                                      DF  11 0024D          BRB     34$                                        ; 0763
                               0000G  CF  DD 0024F 35$:     PUSHL   CURRENT_WINDOW                             ; 0770
                                      01  DD 00253          PUSHL   #1                                         ;
                                      5E  DD 00255          PUSHL   SP                                         ;
                               0000G  CF  9F 00257          PUSHAB  MARK_COMPLETE                              ;
              00000000G  9F           04  FB 0025B 36$:     CALLS   #4, @#SYS$CMKRNL                           ;
                                          04 00262          RET                                                ; 0773

; Routine Size:  611 bytes,    Routine Base:  $CODE$ + 003A
;  463      0774 1 ! The remaining routines must be locked in to the working set as they run
;  464      0775 1 ! at an elevated IPL.
;  465      0776 1
;  466      0777 1 LOCK_CODE;
;  467      0778 1
;  468      0779 1 ROUTINE MARK_CATHEDRAL (WINDOW) : NOVALUE =
;  469      0780 1
;  470      0781 1 !++
;  471      0782 1 !
;  472      0783 1 ! ROUTINE DESCRITION:
;  473      0784 1 !
;  474      0785 1 !       This routine is used to mark the specified window as a Cathedral
;  475      0786 1 !       window.  It must be execured in kernel mode.
;  476      0787 1 !
;  477      0788 1 ! CALLING SEQUENCE:
;  478      0789 1 !       MARK_CATHEDRAL (ARG1)
;  479      0790 1 !
;  480      0791 1 ! INPUT PARAMETERS:
;  481      0792 1 !       ARG1: address of the window to mark
;  482      0793 1 !
;  483      0794 1 ! IMPLICIT INPUTS:
;  484      0795 1 !       none
;  485      0796 1 !
;  486      0797 1 ! OUTPUT PARAMETERS:
;  487      0798 1 !       none
;  488      0799 1 !
;  489      0800 1 ! IMPLICIT OUTPUTS:
;  490      0801 1 !       none
;  491      0802 1 !
;  492      0803 1 ! ROUTINE VALUE:
;  493      0804 1 !       none
;  494      0805 1 !
;  495      0806 1 ! SIDE EFFECTS:
;  496      0807 1 !       none
;  497      0808 1 !
;  498      0809 1 !--
;  499      0810 1
;  500      0811 2 BEGIN
;  501      0812 2
;  502      0813 2 MAP
;  503      0814 2         WINDOW          : REF BBLOCK;           ! address of the window to mark
;  504      0815 2
;  505      0816 2 LOCAL
;  506      0817 2         P               : REF BBLOCK;           ! copy of the window address
;  507      0818 2
;  508      0819 2 P = .WINDOW;                                    ! copy the window address
;  509      0820 2
;  510      0821 2 SET_IPL (IPL$_SYNCH);
;  511      0822 2
;  512      0823 2 IF NOT .P[WCB$V_COMPLETE]
;  513      0824 2 THEN
;  514      0825 3     BEGIN
;  515      0826 3     P[WCB$L_STVBN] = 1;
;  516      0827 3     P[WCB$W_NMAP] = 0;
;  517      0828 2     END;
;  518      0829 2
;  519      0830 2 P[WCB$V_CATHEDRAL] = 1;                 ! mark the window
;   520             0831  2
;   521             0832  2 SET_IPL (0);
;   522             0833  2
;   523             0834  2 RETURN;
;   524             0835  2
;   525             0836  1 END;                                              ! end of routine MARK_CATHEDRAL





                                                                           .PSECT   $LOCKEDC1$,NOWRT,2

                                                      0000 00000 MARK_CATHEDRAL:
                                                                           .WORD    Save nothing                                 ; 0779
                                    50        04   AC   D0 00002           MOVL     WINDOW, P                                    ; 0819
                                    12             08   DA 00006           MTPR     #8, #18                                      ; 0821
                        07     0B   A0             05   E0 00009           BBS      #5, 11(P), 1$                                ; 0823
                               2C   A0             01   D0 0000E           MOVL     #1, 44(P)                                    ; 0826
                                              16   A0   B4 00012           CLRW     22(P)                                        ; 0827
                               0B   A0        40   8F   88 00015 1$:       BISB2    #64, 11(P)                                   ; 0830
                                    12             00   DA 0001A           MTPR     #0, #18                                      ; 0832
                                                        04 0001D           RET                                                   ; 0836

; Routine Size:  30 bytes,    Routine Base:  $LOCKEDC1$ + 0000
```

```
ALL
Sym

DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
DYN
EXE
EXE
FCB
HEA
IND
IO
IP[
IRP
JIB
JIB
MVL
PCB
PR$
RSN
RVT
```

```
;   527         0837  1 ROUTINE ADD_WINDOW (WINDOW, QUEUE_HEAD) : NOVALUE =
;   528         0838  1
;   529         0839  1 !++
;   530         0840  1 !
;   531         0841  1 ! FUNCTIONAL DESCRIPTION:
;   532         0842  1 !
;   533         0843  1 !       This routine adds the window specified into the queue specified. This
;   534         0844  1 !       routine must be called in kernel mode.
;   535         0845  1 !
;   536         0846  1 ! CALLING SEQUENCE:
;   537         0847  1 !       ADD_WINDOW (ARG1, ARG2)
;   538         0848  1 !
;   539         0849  1 ! INPUT PARAMETERS:
;   540         0850  1 !       ARG1: address of the window segement to add
;   541         0851  1 !       ARG2: address of the queue head
;   542         0852  1 !
;   543         0853  1 ! IMPLICIT INPUTS:
;   544         0854  1 !       none
;   545         0855  1 !
;   546         0856  1 ! OUTPUT PARAMETERS:
;   547         0857  1 !       none
;   548         0858  1 !
;   549         0859  1 ! IMPLICIT OUTPUTS:
;   550         0860  1 !       none
;   551         0861  1 !
;   552         0862  1 ! ROUTINE VALUE:
;   553         0863  1 !       none
;   554         0864  1 !
;   555         0865  1 ! SIDE EFFECTS:
;   556         0866  1 !       none
;   557         0867  1 !
;   558         0868  1 !--
;   559         0869  1
;   560         0870  2 BEGIN
;   561         0871  2
;   562         0872  2 MAP
;   563         0873  2         WINDOW          : REF BBLOCK,           ! address of the window segment
;   564         0874  2         QUEUE_HEAD      : REF BBLOCK;           ! address of the queue head
;   565         0875  2
;   566         0876  2 INSQUE (.WINDOW, .QUEUE_HEAD);
;   567         0877  2
;   568         0878  2 RETURN;
;   569         0879  2
;   570         0880  1 END;                                            ! end of routinr ADD_WINDOW



                                        0000 00000 ADD_WINDOW:
                                                        .WORD   Save nothing                              ; 0837
                  08   BC       04   BC  0E 00002       INSQUE  @WINDOW, @QUEUE_HEAD                      ; 0876
                                         04 00007       RET                                               ; 0880

; Routine Size:  8 bytes,    Routine Base:  $LOCKEDC1$ + 001E
;  572          0881  1 ROUTINE REMOVE_WINDOW (WINDOW) : NOVALUE =
;  573          0882  1
;  574          0883  1 !++
;  575          0884  1 !
;  576          0885  1 ! FUNCTIONAL DESCRIPTION:
;  577          0886  1 !
;  578          0887  1 !       This routine removes the specifed window from the queue.  It then
;  579          0888  1 !       proceeds to deallocate the window.  This routine muse be called in
;  580          0889  1 !       kernel mode.
;  581          0890  1 !
;  582          0891  1 ! CALLING SEQUENCE:
;  583          0892  1 !       REMOVE_WINDOW (ARG1)
;  584          0893  1 !
;  585          0894  1 ! INPUT PARAMETERS:
;  586          0895  1 !       ARG1: address of the window to remove
;  587          0896  1 !
;  588          0897  1 ! IMPLICIT INPUTS:
;  589          0898  1 !       none
;  590          0899  1 !
;  591          0900  1 ! OUTPUT PARAMETERS:
;  592          0901  1 !       none
;  593          0902  1 !
;  594          0903  1 ! IMPLICIT OUTPUTS:
;  595          0904  1 !       none
;  596          0905  1 !
;  597          0906  1 ! ROUTINE VALUE:
;  598          0907  1 !       none
;  599          0908  1 !
;  600          0909  1 ! SIDE EFFECTS:
;  601          0910  1 !       none
;  602          0911  1 !
;  603          0912  1 !--
;  604          0913  1
;  605          0914  2 BEGIN
;  606          0915  2
;  607          0916  2 MAP
;  608          0917  2         WINDOW          : REF BBLOCK;           ! address of the window
;  609          0918  2
;  610          0919  2 LOCAL
;  611          0920  2         DUMMY;                                  ! temp storage for queue entry address
;  612          0921  2
;  613          0922  2 EXTERNAL ROUTINE
;  614          0923  2         DEALLOCATE;                             ! deallocate system dynamic memory
;  615          0924  2
;  616          0925  2 REMQUE (.WINDOW, DUMMY);
;  617          0926  2 DEALLOCATE (.WINDOW);
;  618          0927  2
;  619          0928  2 RETURN;
;  620          0929  2
;  621          0930  1 END;                                            ! end of routine REMOVE_WINDOW


                                    0000 00000 REMOVE_WINDOW:
                                                .WORD   Save nothing                          ; 0881
```

```
                         50         04  BC  0F 00002          REMQUE  @WINDOW, DUMMY                          ; 0925
                                    04  AC  DD 00006          PUSHL   WINDOW                                  ; 0926
                  0000G  CF             01  FB 00009          CALLS   #1, DEALLOCATE                          ;
                                            04 0000E          RET                                             ; 0930

; Routine Size:  15 bytes,    Routine Base:  $LOCKEDC1$ + 0026
```

BAD

```
;  623          0931  1 ROUTINE LAST_SEGMENT (WINDOW) : NOVALUE =
;  624          0932  1
;  625          0933  1 !++
;  626          0934  1 !
;  627          0935  1 ! FUNCTIONAL DESCRIPTION:
;  628          0936  1 !
;  629          0937  1 !       This routine zaps the link pointer of the specified window segment
;  630          0938  1 !       therefore making it the last segment in the Cathedral window.
;  631          0939  1 !
;  632          0940  1 ! CALLING SEQUENCE:
;  633          0941  1 !       LAST_SEGMENT (ARG1)
;  634          0942  1 !
;  635          0943  1 ! INPUT PARAMETERS:
;  636          0944  1 !       ARG1: address of the window segment
;  637          0945  1 !
;  638          0946  1 ! IMPLICIT INPUTS:
;  639          0947  1 !       none
;  640          0948  1 !
;  641          0949  1 ! OUTPUT PARAMETERS:
;  642          0950  1 !       none
;  643          0951  1 !
;  644          0952  1 ! IMPLICIT OUTPUTS:
;  645          0953  1 !       none
;  646          0954  1 !
;  647          0955  1 ! ROUTINE VALUE:
;  648          0956  1 !       none
;  649          0957  1 !
;  650          0958  1 ! SIDE EFFECTS:
;  651          0959  1 !       none
;  652          0960  1 !
;  653          0961  1 !--
;  654          0962  1
;  655          0963  2 BEGIN
;  656          0964  2
;  657          0965  2 MAP
;  658          0966  2       WINDOW          : REF BBLOCK;           ! address of the window segment
;  659          0967  2
;  660          0968  2 WINDOW[WCB$L_LINK] = 0;
;  661          0969  2
;  662          0970  2 RETURN;
;  663          0971  2
;  664          0972  1 END;                                          ! end of routine LAST_SEGMENT



                                    0000 00000 LAST_SEGMENT:
                                                      .WORD   Save nothing                          ; 0931
                          50       04   AC  D0 00002   MOVL    WINDOW, R0                            ; 0968
                                   20   A0  D4 00006   CLRL    32(R0)                                ;
                                                04 00009   RET                                       ; 0972

; Routine Size:  10 bytes,    Routine Base:  $LOCKEDC1$ + 0035


;  665          0973  1
```

```
BAD
V04

;
;
;
;
;
;
;
;
```

```
;  666          0974  1 END
;  667          0975  0 ELUDOM



;                                  PSECT SUMMARY
;
;       Name                       Bytes                       Attributes
;
;  $CODE$                              669  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $LOCKEDC1$                           63  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                            Library Statistics
;
;                                                 -------- Symbols --------        Pages         Processing
;       File                                      Total    Loaded   Percent        Mapped        Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1                  18619        36         0        1000            00:01.9



;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:ACPCNTRL/OBJ=OBJ$:ACPCNTRL MSRC$:ACPCNTRL/UPDATE=(ENH$:ACPCNTRL)

; Size:           732 code + 0 data bytes
; Run Time:          00:20.5
; Elapsed Time:      00:52.0
; Lines/CPU Min:       2860
; Lexemes/CPU-Min: 14435
; Memory Used:  231 pages
; Compilation Complete
```

```
BAD
V04

; R
```

0164 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY